Notice of Allowability	**Application No.** 09/976,313	**Applicant(s)** HARRISON ET AL.	
	Examiner Tuan A. Vu	**Art Unit** 2193	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address--

All claims being allowable, PROSECUTION ON THE MERITS IS (OR REMAINS) CLOSED in this application. If not included herewith (or previously mailed), a Notice of Allowance (PTOL-85) or other appropriate communication will be mailed in due course. **THIS NOTICE OF ALLOWABILITY IS NOT A GRANT OF PATENT RIGHTS.** This application is subject to withdrawal from issue at the initiative of the Office or upon petition by the applicant. See 37 CFR 1.313 and MPEP 1308.

1. ☒ This communication is responsive to *1/25/2006*.

2. ☒ The allowed claim(s) is/are *2,6, 8,12, 14,18, 21,26 (renumbered 1-8)*.

3. □ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
   a) □ All b) □ Some* c) □ None of the:
   1. □ Certified copies of the priority documents have been received.
   2. □ Certified copies of the priority documents have been received in Application No. _____ .
   3. □ Copies of the certified copies of the priority documents have been received in this national stage application from the International Bureau (PCT Rule 17.2(a)).

   * Certified copies not received: _____.

Applicant has THREE MONTHS FROM THE "MAILING DATE" of this communication to file a reply complying with the requirements noted below. Failure to timely comply will result in ABANDONMENT of this application.
THIS THREE-MONTH PERIOD IS NOT EXTENDABLE.

4. □ A SUBSTITUTE OATH OR DECLARATION must be submitted. Note the attached EXAMINER'S AMENDMENT or NOTICE OF INFORMAL PATENT APPLICATION (PTO-152) which gives reason(s) why the oath or declaration is deficient.

5. □ CORRECTED DRAWINGS (as "replacement sheets") must be submitted.
   (a) □ including changes required by the Notice of Draftsperson's Patent Drawing Review (PTO-948) attached
   1) □ hereto or 2) □ to Paper No./Mail Date _____.
   (b) □ including changes required by the attached Examiner's Amendment / Comment or in the Office action of Paper No./Mail Date _____.

Identifying indicia such as the application number (see 37 CFR 1.84(c)) should be written on the drawings in the front (not the back) of each sheet. Replacement sheet(s) should be labeled as such in the header according to 37 CFR 1.121(d).

6. □ DEPOSIT OF and/or INFORMATION about the deposit of BIOLOGICAL MATERIAL must be submitted. Note the attached Examiner's comment regarding REQUIREMENT FOR THE DEPOSIT OF BIOLOGICAL MATERIAL.

Attachment(s)

1. □ Notice of References Cited (PTO-892)
2. □ Notice of Draftperson's Patent Drawing Review (PTO-948)
3. □ Information Disclosure Statements (PTO-1449 or PTO/SB/08), Paper No./Mail Date _____
4. □ Examiner's Comment Regarding Requirement for Deposit of Biological Material
5. □ Notice of Informal Patent Application (PTO-152)
6. □ Interview Summary (PTO-413), Paper No./Mail Date _____ .
7. ☒ Examiner's Amendment/Comment
8. ☒ Examiner's Statement of Reasons for Allowance
9. □ Other _____.

DETAILED ACTION

1. This action is responsive to the Applicant's response filed 1/25/2006.

As indicated in Applicant's response, no claims have been amended. Claims 2-6, 8-12, 14-18, 21-25 are pending in the office action.

EXAMINER'S AMENDMENT

2. An examiner's amendment to the record appears below. Should the changes and/or additions be unacceptable to applicant, an amendment may be filed as provided by 37 CFR 1.312. To ensure consideration of such an amendment, it MUST be submitted no later than the payment of the issue fee.

Authorization for this examiner's amendment was given in a telephone interview with Ashley Ott, Reg # 55, 515 on 4/24/2006.

The application has been amended as follows.

In the CLAIMS:

Claim 2:

A computer-implemented method, comprising:

assigning a definition-node for one or more definition statements in an intermediate language program;

assigning a use-node for one or more use statements in the intermediate language program;

performing a memory alias analysis of the intermediate language program to partition memory accesses into equivalence classes such that any two memory accesses that reference the same storage location belong to the same equivalence class;

assigning an alias-node for one or more aliases representing the [[an]] equivalence class of the memory accesses;

introducing an edge into a dependence flow graph connecting each definition-node to the alias-node corresponding to the alias representing the equivalence class to which the definition-node belongs; ~~and~~

introducing an edge in the dependence flow graph connecting each use-node to the alias-node corresponding to the alias representing the equivalence class to which the use-node belongs; and [[,]]

performing a program analysis using the dependence flow graph by assigning, for each alias-node in the dependence flow graph, an initial value to the alias corresponding to said alias-node and adding the alias-node to a set of nodes;

wherein a number of the edges in the dependence flow graph is linear to a number of the nodes in the dependence flow graph, and wherein the number of edges is independent of a definition-use structure of the intermediate language program;

wherein the program analysis further comprises iteratively performing while the set of nodes is not empty:

removing a node from the set of nodes;

if the node is an alias-node, adding successors of the node in the dependence flow graph to the set of nodes; and

if the node is a definition-node for a statement that defines a storage location:

determining a value for an expression to be written to the storage location;

updating the initial value based on the value of the expression; and adding the storage location to the set of nodes.

Claim 3: (Canceled)

Claim 4: (Canceled)

Claim 5: (Canceled)

Claim 6:

The computer-implemented method of claim 2 [[5]], wherein the initial value comprises a set of abstract values which forms a join-complete partial order.

Claim 8:

A machine-readable medium that ~~provides~~ stores instructions, which when executed by a processor, cause the processor to perform operations comprising:

assigning a definition-node for one or more definition statements in an intermediate language program;

assigning a use-node for one or more use statements in the intermediate language program;

performing a memory alias analysis of the intermediate language program to partition memory accesses into equivalence classes such that any two memory accesses that reference the same storage location belong to the same equivalence class;

assigning an alias-node for one or more aliases representing the [[an]] equivalence class of the memory accesses;

introducing an edge into a dependence flow graph connecting each definition-node to the alias-node corresponding to the alias representing the equivalence class to which the definition-node belongs; ~~and~~

introducing an edge in the dependence flow graph connecting each use-node to the alias-node corresponding to the alias representing the equivalence class to which the use-node belongs; and [[,]]

performing a program analysis using the dependence flow graph by assigning, for each alias-node in the dependence flow graph, an initial value to the alias corresponding to said alias-node and adding the alias-node to a set of nodes;

wherein a number of the edges in the dependence flow graph is linear to a number of the nodes in the dependence flow graph, and wherein the number of edges is independent of a definition-use structure of the intermediate language program;

wherein the program analysis further comprises iteratively performing while the set of nodes is not empty:

removing a node from the set of nodes;

if the node is an alias-node, adding successors of the node in the dependence flow graph to the set of nodes; and

if the node is a definition-node for a statement that defines a storage location:

determining a value for an expression to be written to the storage location;

updating the initial value based on the value of the expression; and adding the storage location to the set of nodes.

Claim 9: (Canceled)

Claim 10: (Canceled)

Claim 11: (Canceled)

Claim 12:

The machine-readable medium of claim 8 [[11]], wherein the initial value comprises a set of abstract values which forms a join-complete partial order.

Claim 14:

An apparatus, comprising:

a memory;

a processor coupled to the memory and having a set of instructions which when executed by the processor cause the processor to perform operations comprising:

assigning a definition-node for one or more definition statements in an intermediate language program;

assigning a use-node for one or more use statements in the intermediate language program;

performing a memory alias analysis of the intermediate language program to partition memory accesses into equivalence classes such that any two memory accesses that reference the same storage location belong to the same equivalence class;

assigning an alias-node for one or more aliases representing the [[an]] equivalence class of the memory accesses;

introducing an edge into a dependence flow graph connecting each definition-node to the alias-node corresponding to the alias representing the equivalence class to which the definition-node belongs; ~~and~~

introducing an edge in the dependence flow graph connecting each use-node to the alias-node corresponding to the alias representing the equivalence class to which the use-node belongs; and [[,]]

performing a program analysis using the dependence flow graph by assigning, for each alias-node in the dependence flow graph, an initial value to the alias corresponding to said alias-node and adding the alias-node to a set of nodes;

wherein a number of the edges in the dependence flow graph is linear to a number of the nodes in the dependence flow graph, and wherein the number of edges is independent of a definition-use structure of the intermediate language program;

wherein the program analysis further comprises iteratively performing while the set of nodes is not empty:

removing a node from the set of nodes;

if the node is an alias-node, adding successors of the node in the dependence flow graph to the set of nodes; and

if the node is a definition-node for a statement that defines a storage location:

determining a value for an expression to be written to the storage location;

updating the initial value based on the value of the expression; and adding the storage location to the set of nodes.

Claim 15: (Canceled)

Claim 16: (Canceled)

Claim 17: (Canceled)

Claim 18: The apparatus of claim 14 [[17]], wherein the initial value comprises a set of abstract values which forms a join-complete partial order.

Claim 21:

An computer-implemented apparatus, comprising:

means for assigning a definition-node for one or more definition statements in an intermediate language program;

means for assigning a use-node for one or more use statements in the intermediate language program;

means for performing a memory alias analysis of the intermediate language program to partition memory accesses into equivalence classes such that any two memory accesses that reference the same storage location belong to the same equivalence class;

means for assigning an alias-node for one or more aliases representing the [[an]] equivalence class of the memory accesses;

means for introducing an edge into a dependence flow graph connecting each definition-node to the alias-node corresponding to the alias representing the equivalence class to which the definition-node belongs; ~~and~~

means for introducing an edge in the dependence flow graph connecting each use-node to the alias-node corresponding to the alias representing the equivalence class to which the use-node belongs; and [[,]]

means for performing a program analysis using the dependence flow graph by assigning, for each alias-node in the dependence flow graph, an initial value to the alias corresponding to said alias-node and adding the alias-node to a set of nodes;

wherein a number of the edges in the dependence flow graph is linear to a number of the nodes in the dependence flow graph, and wherein the number of edges is independent of a definition-use structure of the intermediate language program;

wherein the program analysis further comprises iteratively performing while the set of nodes is not empty:

removing a node from the set of nodes;

if the node is an alias-node, adding successors of the node in the dependence flow graph to the set of nodes; and

if the node is a definition-node for a statement that defines a storage location:

determining a value for an expression to be written to the storage location;

updating the initial value based on the value of the expression; and

adding the storage location to the set of nodes.

Claim 22: (Canceled)

Claim 23: (Canceled)

Claim 24: (Canceled)

Claim 25: (Canceled)

Claim 26: (New) The apparatus of claim 21, wherein the initial value comprises a set of abstract values which forms a join-complete partial order.

EXAMINER'S STATEMENT OF REASONS FOR ALLOWANCE

3. Claims 2, 6, 8, 12, 14, 18, 21, and 26 are allowed.

The following is an examiner's statement of reasons for allowance:

The prior art taken separately or jointly does not suggest or teach the following features.

A method for performing a program analysis using the dependence flow graph representing a intermediate language program to partition memory accesses into equivalence classes such that any two memory accesses that reference the same storage location belong to the same equivalence class, the method comprising

(i) assigning a definition-node and a use-node, respectively, for definition statements and use statements in the intermediate language program; assigning an alias-node for one or more aliases representing the equivalence class of the memory accesses; introducing an edge into a dependence flow graph connecting each definition-node to the alias-node corresponding to the alias representing the equivalence class to which the definition-node belongs; and introducing an edge connecting each use-node to the alias-node corresponding to the alias representing the equivalence class to which the use-node belongs; wherein the number of edges is independent of a definition-use structure of the intermediate language program;

(ii) assigning, for each alias-node in the dependence flow graph, an initial value to the alias corresponding to said alias-node and adding the alias-node to a set of nodes; and iteratively performing while the set of nodes is not empty:

removing a node from the set of nodes;

if the node is an alias-node, adding successors of the node in the dependence flow graph to the set of nodes; and

if the node is a definition-node for a statement that defines a storage location:

determining a value for an expression to be written to the storage location;

updating the initial value based on the value of the expression; and

adding the storage location to the set of nodes;

as these limitations are recited in claims 2, 8, 14, and 21.

Cramer, USPN: 5,107,418, teaches extracting alias information for all dependent graph basic block nodes, and iteratively for each node, determining a corresponding alias node; and if such alias node pertains to use-use type of connection, adding that node into the alias equivalent subtree until the next node being reached belongs to a definition type of connection, thus creating a equivalent set of connected alias joined into a use-use class. But Cramer does not suggest or explicit teach the sequences of steps of assigning and joining definition node to alias node corresponding to a same equivalence class; and/or assigning and joining a use-node to a alias node corresponding to a same equivalence class as depicted in (i); nor does Cramer teach the algorithm for adding alias node to a set of nodes with updating the initial value for an expression to be written to the a storage location as depicted in (ii).

Any comments considered necessary by applicant must be submitted no later than the payment of the issue fee and, to avoid processing delays, should preferably accompany the issue fee. Such submissions should be clearly labeled "Comments on Statement of Reasons for Allowance."

Conclusion

4. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Tuan A Vu whose telephone number is (272) 272-3735. The examiner can normally be reached on 8AM-4:30PM/Mon-Fri.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kakali Chaki can be reached on (571)272-3719.

The fax phone number for the organization where this application or proceeding is assigned is (571) 273-3735 (for non-official correspondence – please consult Examiner before using) or 571-273-8300 (for official correspondence) or redirected to customer service at 571-272-3609.

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

VAT
April 25, 2006

KAKALI CHAKI
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100